CodeArts IDE

最佳实践

文档版本 01

发布日期 2026-03-16

华为云计算技术有限公司

版权所有 © 华为云计算技术有限公司 2026。 保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明

HUAWEI和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 最佳实践汇总.....1
2 使用 CodeArts IDE for C/C++ 开发 OpenGL 示例工程.....2
3 使用 CodeArts IDE for Java 开发 Java 示例工程.....6
4 使用 CodeArts IDE for Python 开发简单的 Python 工程.....17

1 最佳实践汇总

在软件开发过程中，开发人员经常需要使用集成开发环境（CodeArts IDE）来编写、调试和测试代码。

当您完成了CodeArts IDE客户端的下载和登录后，可能会遇到不知道如何根据自身的业务需求高效利用CodeArts IDE提供的一系列常用实践的问题。

本文汇总了基于CodeArts IDE常见应用场景的操作实践，为每个实践提供详细的方案描述和操作指导，帮助开发人员快速上手，提高开发效率。

表 1-1 CodeArts IDE 最佳实践一览表

最佳实践	说明
使用CodeArts IDE for C/C++ 开发OpenGL示例工程	本实践向您介绍如何基于CodeArts IDE快速创建简单的C++工程，CodeArts IDE面向开发者提供的智能化可扩展桌面集成开发环境，结合行业和产业开发套件，实现一站式开发体验。
使用CodeArts IDE for Java开发Java示例工程	本实践向您介绍如何基于CodeArts IDE快速创建简单的Java工程，CodeArts IDE for Java是一个JAVA集成开发环境，将文本编辑器和强大的开发工具（如智能代码补全、导航、重构和调试）集成在一起。
使用CodeArts IDE for Python开发简单的Python工程	本实践向您介绍如何基于CodeArts IDE快速创建简单的Python工程，CodeArts IDE带有Python扩展，为Python语言提供了广泛的支持。Python扩展可以利用CodeArts IDE的代码补全、验证、调试和单元测试等特性，与多种Python解释器协同工作，轻松切换虚拟环境。

版权所有 © 华为云计算技术有限公司

2 使用 CodeArts IDE for C/C++ 开发 OpenGL 示例工程

功能介绍

CodeArts IDE面向开发者提供的智能化可扩展桌面集成开发环境（IDE），结合行业和产业开发套件，实现一站式开发体验。

- 编码新体验，开发更高效：内置自研C/C++语言开发支持，提供全新的工程加载、语法着色、符号解析、编码重构和运行调试等开发体验，提升开发效率。
- 能力可扩展，生态更开放：支持基于插件的能力扩展，开放的插件标准，开源的插件框架，开放的插件市场，形成更加开放的生态系统。
- 界面可裁剪，体验更优质：支持基于组件的界面剪裁，在精简模式下形成专用工具的优质体验，又可以在需要时升级为全模式的全量IDE工具。

CodeArts IDE for C/C++：基于C/C++语言开发CMake工程，并通过CodeArts IDE完成从工程创建、代码编写、运行调试到发布测试的全过程。基于插件扩展可以将个人开发者作业流集成其中，实现从需求到提交的全部过程，更可在业务中集成提供的诸多能力，提升应用交付效率。

本实验将指导开发者通过CodeArts IDE for C/C++平台，在本地桌面快速开发一个基于Qt实现的简单项目。通过本实验您将体验到：

- 如何在CodeArts IDE for C/C++上进行基于CMake项目的本地编译构建；
- 在CodeArts IDE上调试和运行；
- 实现一个简单OpenGL demo。

前提条件

- 已下载并安装CodeArts IDE for C/C++。
- 已获取**示例工程代码**。

编译构建与运行调试

通过CodeArts IDE for Cpp客户端以文件夹的形式打开代码包。

步骤1 **登录到CodeArts IDE for C/C++客户端**，单击“构建”，选择“CMake构建类型 > Debug”。

版权所有 © 华为云计算技术有限公司

步骤2 导入样例代码，然后在弹窗中选择“all META”，单击右上角构建按钮。

步骤3 当编译构建成功后，日志打印finished，并显示构建成功的图示。

步骤4 构建成功后，在页面的右上角选择“OpenGLSample”，然后单击右侧的运行按钮。

----结束

版权所有 © 华为云计算技术有限公司

关键代码片段详解

```
// MyWidget类的构造函数，初始化一些成员变量并设置布局
MyWidget::MyWidget(QWidget *parent) : QWidget(parent)
{
    resize(800, 600);// 设置窗口大小为800*600

   // 创建一个垂直布局，用于放置菜单标题和按钮
   m_menuButtonLayout = new QVBoxLayout;
   m_menuButtonLayout->setAlignment(Qt::AlignTop); // 设置布局的对齐方式为顶部对齐
   // 创建一个标签，设置文本为"Menu"，并设置其居中对齐
   m_menuTitle = new QLabel(this);
   m_menuTitle->setText("Menu");
   m_menuTitle->setAlignment(Qt::AlignCenter);
   // 创建一个按钮，设置文本为"OpenGL Sample"，并设置其固定大小
   m_openGLDemoButton = new QPushButton("OpenGL Sample", this);
   m_openGLDemoButton->setFixedSize(140, 30);
   // 将标签和按钮添加到垂直布局中
   m_menuButtonLayout->addWidget(m_menuTitle);
   m_menuButtonLayout->addWidget(m_openGLDemoButton);
   // 创建一个新的窗口部件，并设置其布局为上面创建的垂直布局
   m_menuWidget = new QWidget(this);
   m_menuWidget->setLayout(m_menuButtonLayout);
   // 创建一个帧，设置其形状为垂直线并设置阴影效果
   QFrame* line = new QFrame(this);
   line->setFrameShape(QFrame::VLine);
   line->setFrameShadow(QFrame::Sunken);
   // 创建一个水平布局，用于放置菜单窗口部件和帧
   QHBoxLayout *mainLayout = new QHBoxLayout;
   mainLayout->addWidget(m_menuWidget, 1); // 将菜单窗口部件添加到布局中，并设置其伸缩因子为1
   mainLayout->addWidget(line); // 将帧添加到布局中
   setLayout(mainLayout); // 将主布局设置为窗口的布局
   // 连接按钮的单击信号到槽函数ShowOpenGLDemo
   connect(m_openGLDemoButton, &QPushButton::clicked, this, &MyWidget::ShowOpenGLDemo);
}
// 槽函数，用于显示OpenGL演示
void MyWidget::ShowOpenGLDemo()
{
   OpenGLDemoShow(); // 调用OpenGL演示显示函数
   return;
}
```


版权所有 © 华为云计算技术有限公司

查看运行结果

单击“OpenGL Sample”，结果如下所示：

版权所有 © 华为云计算技术有限公司

3 使用 CodeArts IDE for Java 开发 Java 示例工程

功能介绍

CodeArts IDE for Java是一个JAVA集成开发环境，将文本编辑器和强大的开发工具（如智能代码补全、导航、重构和调试）集成在一起。

- 调试热替换：支持在调试模式下运行程序。当程序挂起时，在“运行”和“调试”视图中检查其输出。找到错误并更正它，然后重新运行程序。在Java上下文中，可通过热代码替换功能来动态修改和重新加载类，无需停止调试会话。
- 强大的编码辅助能力：CodeArts IDE for Java可以深入理解代码，提供上下文感知的机器学习辅助补全功能，可以补全单条语句或整行代码。对于代码编辑器中的任何符号，开发者可以查看其定义和相关文档。集成开发环境可高亮显示错误，并让开发者直接在代码编辑器中对错误采取行动。“问题”视图会列出当前打开的文件中检测到的所有问题。
- Java构建工具集成：CodeArts IDE for Java提供对Maven和Gradle构建系统的内置支持，包括项目和依赖关系解析，以及运行Maven目标和Gradle任务的能力。

Spring Boot和Thymeleaf是目前非常流行的Java开发框架和模板引擎。Spring Boot提供了快速构建Web应用程序的能力，而Thymeleaf则是一种功能强大且易于使用的模板引擎，可以帮助在服务器端生成动态的HTML页面。在本文中，将使用Spring Boot和Thymeleaf来实现一个完整的用户管理功能，包括增加、删除、修改和查询用户信息。 以及使用CodeArts IDE for Java构建，运行Java工程，并且学到Java语言的一些基础知识，例如Java的一些基本类型。通过Java语言创建，读取和修改本地文件的能力。

前置条件

- 已下载并安装CodeArts IDE for Java，了解IDE的基础功能。
- 已获取示例工程代码。
- 安装Java JDK 17及其以上版本。
- 已引入spring-boot-starter-web、spring-boot-starter-thymeleaf和lombok（可选）相关依赖。

版权所有 © 华为云计算技术有限公司

开发时序图

创建工程

步骤1 **登录到CodeArts IDE for Java客户端**，选择“新建工程”创建Java工程。

其中“构建系统”选择“Maven”，“框架”选择“SpringBoot”。

版权所有 © 华为云计算技术有限公司

步骤2 单击“创建”，项目开始加载，会启动相关服务、下载依赖的Jar包等，此过程受计算机性能、网速等因素影响会耗费一定的时间。

步骤3 Java工程创建完成后，在编辑器界面，单击“文件 > 打开项目”，导入示例工程代码。

----结束

运行调试

步骤1 在具有“main()”方法的“UserManagerApplication”类的代码编辑器中，单击或右键“main()”方法左侧**“运行/调试主类”**按钮（ ）。

步骤2 在弹出的下拉列表中，单击“在UserManagerApplication中运行main方法”或“在UserManagerApplication中调试main方法”项，将运行或调试项目。如下图所示：

版权所有 © 华为云计算技术有限公司

图 3-1 代码编辑区区域“main()”方法处启动调试会话入口

步骤3 在浏览器地址栏输入“localhost:8080”。

----结束

输出结果示例

程序运行截图如下：

关键代码片段

以下为关键代码片段，全量示例代码请单击示例工程代码获取。

后端实现过程

pojo类：（@Data是lombod注解，不需要写get、set方法了。）

```
@Data
public class User {
    private String uuid;
    private Integer id;
    private String name;
```

```
    private Integer age;
    private String department;
    public User(){}
    public User(Integer id, String name, Integer age, String department) {
        this.uuid = UUID.randomUUID().toString();
        this.id = id;
        this.name = name;
        this.age = age;
        this.department = department;
    }
}
```

mapper接口：

```
public interface IUserMapper {
    public List<User> getUserList();
    public boolean addUser(User user);
    public boolean updateUser(User user);
    public boolean deleteUser(String uuid);
}
```

mapperImpl实现类：

```
@Component
public class UserMapperImpl implements IUserMapper {
    @Override
    public List<User> getUserList() {
        return UserData.userList;
    }
    @Override
    public boolean addUser(User user) {
        user.setUuid(UUID.randomUUID().toString());
        return UserData.userList.add(user);
    }
    @Override
    public boolean updateUser(User user) {
        Stream<User> userStream = UserData.userList.stream().filter(userItem -> {
            if (user.getId().equals(userItem.getId())) {
                if (user.getId() != null) {
                    userItem.setId(user.getId());
                }
                if ( !StringUtils.isEmpty(user.getName())) {
                    userItem.setName(user.getName());
                }
                if (!StringUtils.isEmpty(user.getDepartment())) {
                    userItem.setDepartment(user.getDepartment());
                }
                if (!StringUtils.isEmpty(user.getAge())) {
                    userItem.setAge(user.getAge());
                }
                return true;
            }
            return false;
        });
        return userStream.count() >= 1;
    }
    @Override
    public boolean deleteUser(String uuid) {
        int sizeBeforeDelete = UserData.userList.size();
        List<User> deletedUsers = UserData.userList.stream().filter(user ->
uuid.equals(user.getUuid())).collect(Collectors.toList());
        deletedUsers.forEach(UserData.userList::remove);
        return sizeBeforeDelete != UserData.userList.size();
    }
}
```

service接口：

```
public interface IUserService {
    public List<User> getUserList();
```


版权所有 © 华为云计算技术有限公司

```
    public boolean addUser(User user);
    public boolean updateUser(User user);
    public boolean deleteUser(String id);
}
```

serviceImpl实现类：

```
@Service
public class UserServiceImpl implements IUserService {
    @Autowired
    IUserMapper userMapperImpl;
    @Override
    public List<User> getUserList() {
        return userMapperImpl.getUserList();
    }
    @Override
    public boolean addUser(User user) {
        return userMapperImpl.addUser(user);
    }
    @Override
    public boolean updateUser(User user) {
        return userMapperImpl.updateUser(user);
    }
    @Override
    public boolean deleteUser(String id) {
        return userMapperImpl.deleteUser(id);
    }
}
```

controller控制层： 用户管理接口Controller。

```
@RestController
@RequestMapping("/user")
public class UserController {
    @Autowired
    IUserService userServiceImpl;
    @GetMapping("/getUserList")
    public List<User> getUserList() {
        return userServiceImpl.getUserList();
    }
    @PostMapping("/addUser")
    public boolean addUser(User user) {
        return userServiceImpl.addUser(user);
    }
    @PostMapping("/updateUser")
    public boolean updateUser(User user) {
        return userServiceImpl.updateUser(user);
    }
    @PostMapping("/deleteUser")
    public boolean deleteUser(String uuid) {
        return userServiceImpl.deleteUser(uuid);
    }
}
```

用户欢迎界面Controller：

```
@Controller
public class WelcomePageController {
    @RequestMapping("/")
    public String user() {
        return "user";
    }
}
```

用户数据：

```
public class UserData {
    public static List<User> userList = new ArrayList<>();

    public static void initUserData() {
```


版权所有 © 华为云计算技术有限公司

```
        userList.add(new User(1, "张三", 20, "人力资源部"));
        userList.add(new User(2, "李四", 18, "后勤部"));
        userList.add(new User(3, "王五", 21, "研发部"));
        userList.add(new User(4, "赵六", 25, "产品部"));
    }
}
```

启动类：（每次启动调试需要初始化用户数据）

```
@SpringBootApplication
public class UserManagerApplication {

    public static void main(String[] args) {
        UserData.initUserData(); // 初始化用户数据
        SpringApplication.run(UserManagerApplication.class, args);
    }

}
```

前端实现过程(thymeleaf模板+html+ajax)

user.html：

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <title>User Manager</title>
    <script src="https://cdn.bootcdn.net/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
    <script>
        $(function () {
            queryData();
        });

        function queryData() {
            // 发送 AJAX 请求
            $.ajax({
                url: 'http://localhost:8080/user/getUserList',
                type: 'GET',
                dataType: 'json',
                success(res) {
                    var html = '';
                    for (var i = 0; i < res.length; i++) {
                        html += '<tr><td>' + res[i].id + '</td>' + '<td>' + res[i].name + '<td>' + res[i].age + '<td>' +
res[i].department + '</td><td> <button onclick="del(\'' + res[i].uuid + '\')">删除</button></td></tr>';
                    }
                    // console.log(html);
                    $('#tt').html(html);
                },
                error: function (xhr, status, error) {
                    console.error(error); // 处理错误情况
                }
            });
        }

        function add() {
            const id = document.getElementById('InputId').value;
            const name = document.getElementById('InputName').value;
            const age = document.getElementById('InputAge').value;
            const department = document.getElementById('InputDepartment').value;

            if (!validateId(id)) {
                return;
            }

            $.ajax({
                url: `http://localhost:8080/user/addUser?id=${id}&name=${name}&age=${age}&department=$
{department}`,
                type: 'post',
```


版权所有 © 华为云计算技术有限公司

```
            dataType: 'json',
            success: function (res) {
                queryData();
                if (res == true) {
                    alert('添加成功');
                } else {
                    alert('添加失败');
                }
            },
            error: function (xhr, status, error) {
                console.error(error);
            }
        });
        document.getElementById('InputId').value = '',
            document.getElementById('InputName').value = '';
        document.getElementById('InputAge').value = '';
        document.getElementById('InputDepartment').value = '';
    }

    function update() {
        const id = document.getElementById('updateInputId').value;
        const name = document.getElementById('updateInputName').value;
        const age = document.getElementById('updateInputAge').value;
        const department = document.getElementById('updateInputDepartment').value;

        if (!validateId(id)) {
            return;
        }

        $.ajax({
            url: `http://localhost:8080/user/updateUser?id=${id}&name=${name}&age=${age}&department=$
{department}`,
            type: 'post',
            dataType: 'json',
            success: function (res) {
                queryData();
                if (res == true) {
                    alert('修改成功');
                } else {
                    alert('修改失败');
                }
            },
            error: function (xhr, status, error) {
                console.error(error);
            }
        });
        document.getElementById('updateInputId').value = '',
            document.getElementById('updateInputName').value = '';
        document.getElementById('updateInputAge').value = '';
        document.getElementById('updateInputDepartment').value = '';
    }

    function del(uuid) {
        $.ajax({
            url: `http://localhost:8080/user/deleteUser?uuid=${uuid}`,
            type: 'post',
            dataType: 'json',
            success: function (res) {
                queryData();
                if (res == true) {
                    alert('删除成功');
                } else {
                    alert('删除失败');
                }
            },
            error: function (xhr, status, error) {
                console.error(error);
            }
        });
```


版权所有 © 华为云计算技术有限公司

```
        }

        function validateId(id) {
            if (id == '') {
                alert("Id值不能为空");//空值校验
                return false;
            }
            return true;
        }
    </script>
</head>

<body>
    <div class="header">
        <!--<button onclick="queryData()">查询</button>-->
        <div class="items">
            <input type="number" value="" placeholder="请输入id" id="InputId" />
            <input type="text" value="" placeholder="请输入名称" id="InputName">
            <input type="number" value="" placeholder="请输入年龄" id="InputAge">
            <input type="text" value="" placeholder="请输入部门" id="InputDepartment">
            <button onclick="add()">添加</button>
        </div>
        <div class="items">
            <input type="number" value="" placeholder="请输入id" id="updateInputId" />
            <input type="text" value="" placeholder="请输入名称" id="updateInputName">
            <input type="number" value="" placeholder="请输入年龄" id="updateInputAge">
            <input type="text" value="" placeholder="请输入部门" id="updateInputDepartment">
            <button onclick="update()">修改</button>
        </div>
    </div>
    <div class="table">
        <table>
            <thead>
                <tr>
                    <th>用户Id</th>
                    <th>姓名</th>
                    <th>年龄</th>
                    <th>部门</th>
                    <th>操作</th>
                </tr>
            </thead>
            <tbody id="tt">
            </tbody>
        </table>
    </div>
</body>
<style>
    table {
        margin: 20px;
        border-collapse: collapse;
    }

    th {
        background-color: #cccccc;
    }

    tr {
        text-align: center;
        border: 1px solid #ccc;
    }

    td {
        padding: 20px;
        border: 1px solid #ccc;
    }

    button {
        background-color: white;
        width: 70px;
```


版权所有 © 华为云计算技术有限公司

```
        border-radius: 20px;
        border: 1px solid #ccc;
    }

    button :hover {
        background-color: #cccccc;
    }

    .header {
        display: flex;
        justify-content: center;
        flex-wrap: wrap;
    }

    .items {
        display: grid;
        margin-left: 10px;
    }

    .table {
        display: flex;
        justify-content: center;
    }
</style>

</html>
```

相关配置文件

pom.xml：

相关的依赖信息如下，引入spring-boot-starter-web、spring-boot-starter-thymeleaf和lombok（可选）相关依赖。

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
                             http://maven.apache.org/xsd/maven-4.0.0.xsd">

    <modelVersion>4.0.0</modelVersion>

    <groupId>com.huawei</groupId>
    <artifactId>codeartside-user-manager</artifactId>
    <version>1.0-SNAPSHOT</version>

    <!-- Spring Boot parent 管理版本 -->
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.7.14</version>
        <relativePath/>
    </parent>

    <properties>
        <java.version>1.8</java.version>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    </properties>

    <dependencies>
        <!-- Web -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <!-- 模板引擎 -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-thymeleaf</artifactId>
```


版权所有 © 华为云计算技术有限公司

```
        </dependency>

        <!-- Lombok -->
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <scope>provided</scope>
        </dependency>

        <!-- 测试 -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <!-- 编译器插件 -->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.11.0</version>
                <configuration>
                    <source>${java.version}</source>
                    <target>${java.version}</target>
                    <encoding>${project.build.sourceEncoding}</encoding>
                </configuration>
            </plugin>
        </plugins>
    </build>

</project>
```

application.properties：

```
spring.application.name=user-manager
server.port=8080
```

application.yml：

```
thymeleaf:
  prefix:
    classpath: /templates   # 访问template下的html文件需要配置模板，映射
```

代码解析

在html代码中，主要利用onclick单击事件和jquery框架中的ajax交互。 jquery中ajax的语法格式如下：

```
$.ajax({
     url: 接口名,
     type: 'get',
     dataType: 'json',
     success: function(res) {
         queryData();
     },
     error: function(xhr, status, error) {
         console.error(error); // 处理错误情况
    }
 });
```


版权所有 © 华为云计算技术有限公司

4 使用 CodeArts IDE for Python 开发简单的 Python 工程

功能介绍

CodeArts IDE for Python是一款专为Python开发设计的集成开发环境（IDE），旨在提供强大的编辑、调试和项目管理功能，以提高开发效率。CodeArts IDE带有Python扩展，为Python语言提供了广泛的支持。

Python扩展可以利用CodeArts IDE的代码补全、验证、调试和单元测试等特性，与多种Python解释器协同工作，轻松切换虚拟环境。

前提条件

- 已下载并安装CodeArts IDE for Python。
- 已安装Python解释器。

操作步骤

步骤1 **登录到CodeArts IDE for Python客户端**，选择“文件 > 新建 > 工程”，创建Python工程。

在打开的“新建工程”对话框中，选择左侧列表“Python”。

版权所有 © 华为云计算技术有限公司

依次填写以下信息：

1. 填写工程的名称和保存路径。
2. 勾选“新环境使用”中下拉框的“Virtualenv”选项。
3. 在“基础解释器”列表中选择已有解释器。通常CodeArts IDE会自动从标准安装位置检测解释器位置并展示在此。
4. 勾选“创建main.py示例脚本”，以便CodeArts IDE可以使用示例内容填充项目，以便快速使用IDE的主要功能。

步骤2 单击“创建”，CodeArts IDE将创建并打开项目。在项目根目录下的“venv”文件夹中创建一个新环境，并将其设置为项目解释器。

步骤3 双击main.py文件，打开编辑器。输入以下简单的Python代码示例：

```
def main():
    print("Hello, World!")

if __name__ == "__main__":
    main()
```

步骤4 编辑并运行代码。

版权所有 © 华为云计算技术有限公司

1. 在编写代码时，CodeArts IDE会提供代码补全建议。代码补全功能会在您键入时自动显示关键词信息。

2. 代码编写完成后，单击菜单中的“文件 > 保存”。
3. 在项目资源管理器中，右键单击main.py文件，选择“运行 Python 文件”。

4. 运行结果会显示在底部的“调试控制台”中，会看到**Hello, World!**的输出。

版权所有 © 华为云计算技术有限公司

步骤5 调试代码

1. CodeArts IDE的Python扩展提供了调试支持，让您能够设置断点、检查数据，在逐步执行程序时使用调试控制台。将光标悬置在某行代码上，通过单击编辑器左侧行号旁的空白区域来设置断点（会出现一个红点）。

2. 单击CodeArts IDE工具栏上的“**开始调试**”按钮（ ），调试器将在包含断点的那一行暂停运行。

   此时可以使用调试工具栏上的命令来控制程序执行，并查看“运行和调试”视图中的“变量”来检查变量。

步骤6 创建和运行一个unittest测试的示例。

1. 创建一个测试对象，即新建一个名为“inc_dec.py”的文件，它包含以下内容：

```
def increment(x):
    return x + 1
def decrement(x):
   return x - 1
```

2. 创建一个 “unittest” 测试来覆盖这个测试对象，即创建一个名为“inc_decunittest.py” 的文件，包含以下内容：

```
import inc_dec    # The code to test
import unittest   # The test framework
class Test_TestIncrementDecrement(unittest.TestCase):
    def test_increment(self):
```


版权所有 © 华为云计算技术有限公司

```
        self.assertEqual(inc_dec.increment(3), 4)
    def test_decrement(self):
        self.assertEqual(inc_dec.decrement(3), 4)
if __name__ == '__main__':
    unittest.main()
```

步骤7 在CodeArts IDE中配置unittest框架集成。

1. 单击CodeArts IDE底部面板中的“**测试**”，打开“测试”视图，单击“**配置Python测试**”。

2. 在弹出的对话框中，选择您想要启用集成的测试框架，在本例中选择“unittest”。

3. 在新的对话框中，选择包含测试源文件的项目文件夹。在本例中选择项目根文件夹（“.”）。

4. 选择用于标识测试文件的文件通配符模式，本例中为*test.py。

版权所有 © 华为云计算技术有限公司

步骤8 在配置完框架集成后，CodeArts IDE会自动检测测试并在“测试”视图中显示它们。

步骤9 运行和调试测试，包括运行单个测试文件和单个方法。

----结束

版权所有 © 华为云计算技术有限公司